# What's This Movie About? A Joint Neural Network Architecture for Movie Content Analysis

# What's this Movie about?
# A Joint Neural Network Architecture for Movie Content Analysis

**Philip John Gorinski** and **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh, EH8 9AB
P.J.Gorinski@sms.ed.ac.uk, mlap@inf.ed.ac.uk

## Abstract

This work takes a first step toward movie content analysis by tackling the novel task of movie overview generation. Overviews are natural language texts that give a first impression of a movie, describing aspects such as its genre, plot, mood, or artistic style. We create a dataset that consists of movie scripts, attribute-value pairs for the movies' aspects, as well as overviews, which we extract from an online database. We present a novel end-to-end model for overview generation, consisting of a multi-label encoder for identifying screenplay attributes, and an LSTM decoder to generate natural language sentences conditioned on the identified attributes. Automatic and human evaluation show that the encoder is able to reliably assign good labels for the movie's attributes, and the overviews provide descriptions of the movie's content which are informative and faithful.

## 1 Introduction

Movie summarization is the task of automatically summarizing a screenplay in order to gain a general impression of its content. This may include describing the movie's main characters and plot, its genre, artistic style, and so on. As more and more movies are being produced every year[1], there is an ever growing need to facilitate this task. Potential applications include producing shorter versions of scripts to help with the decision making process in a production company, enhancing movie search by generating descriptions of what the movie is about, and notably, supporting movie recommendation engines by abstracting over specific keywords to more general concepts.

Figure 1 gives an example of the type of movie content analysis we would like to obtain automatically. The information is taken from Jinni, a large database (and movie recommendation engine) which indexes movies based on attributes and their values[2] (see the top half of Figure 1) and further aggregates these into a comprehensive overview (see the second half of Figure 1). Jinni's movie attributes were created by film professionals based on analysis of user reviews and metadata. There are hundreds, and they aim to describe aspects such as mood, style, plot, and setting for any released movie or TV show. Although some of these attributes could not be possibly ascribed without information from external sources (e.g., *Praise*, or *Based on*), others could be inferred by watching the movie or reading the

| | |
|---|---|
| *Mood:* | Suspenseful, Captivating, Tense, Scary |
| *Plot:* | Serial Killer, Special Agents, Investigation, Mind Game, Psychopath, Crimes, Deadly, Law Enforcement, Mind and Soul, Rivalry |
| *Genre:* | Crime, Thriller |
| *Style:* | Strong Female Presence |
| *Attitude:* | Serious, Realistic |
| *Place:* | Maryland, USA, Virginia |
| *Period:* | 20th Century, 90s |
| *Based on:* | Based on Book |
| *Praise:* | Award Winner, Blockbuster, Critically Acclaimed, Oscar Winner, Modern Classic, Prestigious Awards |
| *Flag:* | Brief Nudity, Sexual Content, Strong Violent Content |

The Silence of the Lambs can be described as tense, captivating, and suspenseful. The plot revolves around special agents, mind games, and a psychopath. The main genres are thriller and crime. In terms of style, The Silence of the Lambs stars a strong female character. In approach, it is serious and realistic. It is located in Maryland and Virginia. The Silence of the Lambs takes place in the 1990s. It is based on a book. The movie has received attention for being a modern classic, an Oscar winner, and a blockbuster. Note that The Silence of the Lambs involves brief nudity and sexual content.

Figure 1: Jinni attributes, their values, and overview for "The Silence of the Lambs". Underlined attribute values appear in the overview.

[1] According to http://www.boxofficemojo.com/ during 2009–2016, movie releases went up from 536 to 729.

[2] Throughout this paper *attributes* are in italic font and their values in sans serif.

screenplay (e.g., *Genre, Plot, Flag, Mood, Place*).

This work takes a step toward automatic script summarization by jointly modeling the tasks of movie attribute identification and overview generation. Specifically, we propose a novel neural network architecture which draws insights from encoder-decoder models recently proposed for machine translation (Bahdanau et al., 2015) and related sentence generation tasks (Wen et al., 2015; Mei et al., 2016; Lebret et al., 2016). Our model takes the screenplay as input and generates an overview for it. Rather than representing the script as a sequence, we employ feed-forward neural networks (Zhang and Zhou, 2006; Kurata et al., 2016) to encode the screenplay into various attributes (e.g., *Plot*, *Genre*) and their labels (e.g., thriller, romance), viewing movie content analysis as a multi-label classification problem. Our decoder generates movie overviews using a Long Short-Term Memory network (LSTM; Hochreiter and Schmidhuber, 1997), a type of recurrent neural network with a more complex computational unit which is semantically conditioned (Wen et al., 2015, 2016) on this attribute specific representation. Our model is trained end-to-end using screenplays and movie overviews as the supervision signal.

In both automatic and human-based evaluations our neural network architecture outperforms competitive baselines and generates movie overviews which are well-received by human judges. To the best of our knowledge, this is the first work to automatically analyze and summarize the content of screenplays.

## 2 Related Work

Recent years have seen increased interest in the computational analysis of movie screenplays. Ye and Baldwin (2008) create animated storyboards using the action descriptions of movie scripts. Danescu-Niculescu-Mizil and Lee (2011) use screenplays to study the coordination of linguistic styles in dialog. Bamman et al. (2013) induce personas of film characters from movie plot summaries. Agarwal et al. (2014a; 2014b; 2015) extract social networks from scripts, create *xkcd* movie narrative charts, and automate the Bechdel test which is designed to assess the presence of women in movies. Gorinski and Lapata (2015) summarize screenplays by selecting important scenes. Our work joins this line of research in an attempt to automatically induce information pertaining to a movie's content such as its genre and plot elements.

There has been a surge of interest recently in repurposing sequence transduction neural network architectures for various generation tasks such as machine translation (Sutskever et al., 2014), sentence compression (Chopra et al., 2016), and simplification (Zhang and Lapata, 2017). Central to these approaches is an encoder-decoder architecture modeled by recurrent neural networks. The encoder reads the source sequence into a list of continuous-space representations from which the decoder generates the target sequence. Previously proposed architectures are not directly applicable to our task for at least two reasons: (a) the correspondence between screenplays and overviews is very loose, and (b) the screenplay is not strictly speaking a sequence (a screenplay is more like a book consisting of thousands of sentences), and cannot be easily compressed into a vector-based representation from which to generate the overview.

Rather than attempting to decode the overview directly from the screenplay, we encode the latter into attribute-value pairs which we then decode into overviews. We conceptualize the generation task as a joint problem of multi-label categorization, where each screenplay is assigned to one or more categories, and content-sensitive natural language generation. Many machine learning techniques have been proposed for building automatic text categorization systems (see Sebastiani, 2002 and Dalal and Zaveri, 2011 for overviews), including neural networks (Belanger and McCallum, 2016; Kurata et al., 2016). Our encoder is a feed-forward neural network, which, however, is able to capture label interactions which are important for our content analysis task. Our decoder employs an enhanced LSTM architecture which directly maximizes the probability of the overview given the screenplay's attribute values. Conditional LSTMs have been applied to various related tasks, including image description generation (Vinyals et al., 2015), the verbalization of database records (Mei et al., 2016; Lebret et al., 2016), and the generation of dialogue acts (Wen et al., 2015, 2016).

## 3 The Jinni Movie Dataset

Our dataset was built on top of ScriptBase, a collection of 1,276 movie scripts, which Gorinski and

| Attribute | Jinni | Frequent | Merged |
|---|---|---|---|
| Mood | 29 | 19 | 19 |
| Plot | 406 | 101 | 101 |
| Genre | 31 | 31 | 31 |
| Attitude | 8 | 8 | 8 |
| Place | 173 | 53 | 24 |
| Flag | 9 | 9 | 6 |

Table 1: Movie attributes and their values.

Lapata (2015) obtained by automatically crawling web-sites such as imsdb.com. We crawled Jinni[3] in order to obtain attributes and overviews (see Figure 1) for each movie in ScriptBase. As mentioned earlier, attributes have values which are essentially labels/tags describing the movie's content, whereas overviews are short summaries giving a first impression of the movie. The crawl resulted in 917 movies which Jinni and ScriptBase had in common. We further split these into training, development and test sets, with 617, 200, and 100 instances, respectively. We concentrate on the six types of attributes shown in Table 1 whose values we hypothesize can be inferred from analyzing the movie's screenplay.

Table 1 provides an overview of the number of labels used in our experiments. Jinni contains a wealth of attribute values varying from nine for *Flag* to more than 400 for *Plot*. Additionally, value names for some attributes are synonyms or near-synonyms (e.g., Nudity and Brief Nudity for *Flag*). We reduced the set of attribute values to those that occurred most frequently (column Frequent in the table) and merged synonyms into a common label (column Merged).

## 4 Neural Network Architecture

We could approach the movie overview generation task using an attention-based encoder-decoder model (Bahdanau et al., 2015). The encoder would transform the screenplay into a sequence of hidden states with an LSTM (Hochreiter and Schmidhuber, 1997) or another type of computational unit (Cho et al., 2014). The decoder would use another recurrent neural network to generate the overview one word at time, conditioning on all previously generated words and the representation of the input, while an attention mechanism would revisit the input sequence dynamically highlighting pieces of information relevant for the generation task. As mentioned earlier, viewing screenplays as a sequence of sentences is problematic both computationally and conceptually. Even if we used a hierarchical encoder (Tang et al., 2015; Yang et al., 2016) by first building representations of sentences and then aggregating those into a representation of a screenplay, it is doubtful whether a fixed length vector could encode the content of the movie in its entirety or whether the attention mechanism would effectively isolate the parts of the input relevant for generation.

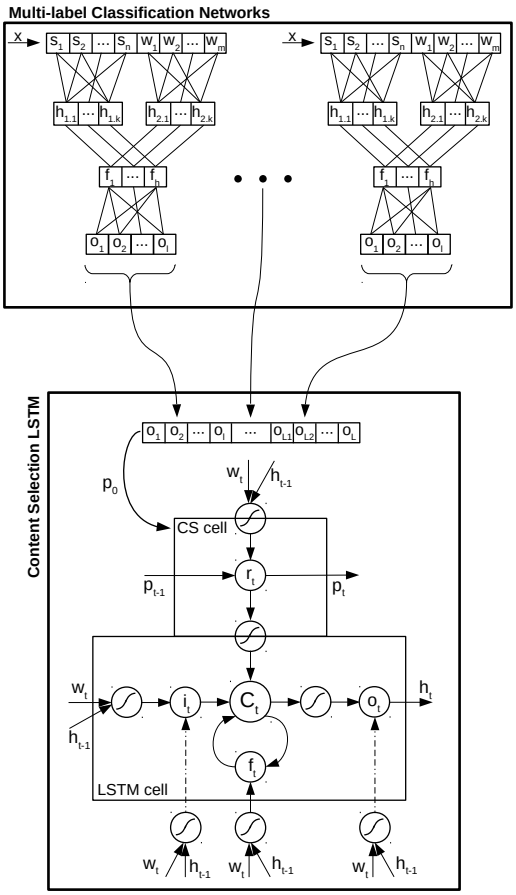


Figure 2: Neural network architecture: given feature vector $x$ representing a screenplay, we employ feed-forward multi-label classification networks to encode the movie into a content vector $p_\theta$ representing attribute labels; this encoding is fed into an LSTM with a content selection cell.

We therefore propose an architecture that consists of two stacked neural network models for the tasks of movie attribute identification and overview generation. Figure 2 illustrates our model. We use simple feed-forward neural networks to impose some structure on the input by identifying the labels that most likely apply to the screenplay. We subsequently employ a semantically conditioned LSTM (Wen et al., 2015, 2016)

[3]Dated on 2015/04/18 and available from http://www.jinni.com/.

to select the content for which to generate sentences. This architecture is advantageous for a number of reasons. Firstly, by imposing structure on the screenplays, the generation network is faced with a more compact and informative representation. This allows us to make use of a content selection LSTM similar to Wen et al., (2015; 2016), generating fluent and label-specific outputs. Secondly, it enables us to train the screenplay encoder (aka classification network) and the decoder jointly, in an end-to-end fashion.

### 4.1 Multi-label Encoder

As shown in Figure 1, the overview highlights various aspects of the movie, essentially devoting a sentence to each attribute. This observation motivates us to encode the screenplay as a set of attributes (with their values) and then decode these into a sentence one by one. We treat attribute encoding as a multi-label classification problem: an attribute (e.g., *Genre* or *Plot*), will typically have multiple values (aka labels) which are suitable for the movie and should occur in the generated sentence. Furthermore, these labels naturally influence each other. For example, a movie whose *Genre* is Crime is also likely to be a Thriller while it is less likely to be a Parody. In traditional multi-label classification such interactions are either ignored (Read et al., 2011; Tsoumakas and Katakis, 2006; Godbole and Sarawagi, 2004; Zhang and Zhou, 2005), or represented by label combinations (Tsoumakas and Vlahavas, 2007; Read et al., 2008). A few approaches assume or impose an existing structure on the label space (Schwing and Urtasun, 2015; Chen et al., 2015; Huang et al., 2015; Jaderberg et al., 2014; Stoyanov et al., 2011; Hershey et al., 2014; Zheng et al., 2015).

We employ a neural network approach with the aim of abstracting the screenplay into a set of meaningful labels whose correlations are discovered automatically, during training. As shown on top of Figure 2, our encoder is a feed-forward neural network where individual neurons represent the labels to be classified. The input to the network is a feature vector $x$ representing the screenplay (we discuss the specific features we use in more detail shortly):

$$h_n = \sigma(W_n x_n) \tag{1}$$

The input is split into $k$ segments by feature type, and the feature segments are fed into $k$ separate fully connected hidden layers. The hidden layer outputs are then combined using simple element-wise addition:

$$f = h_1 \oplus h_2 \oplus \cdots \oplus h_k \tag{2}$$

The combined feature layer is used to compute an $l$-sized output layer, where $l$ corresponds to the size of the classification label set. The final activation of the output units is obtained by applying the sigmoid function to the output layer:

$$O = \sigma(W_o f) \tag{3}$$

In order to better capture label interactions, we adapt a method of network initialization recently introduced in Kurata et al. (2016). In this approach, instead of initializing the model's output weights $W_o$ from a uniform distribution, the first $p$ rows of the weight matrix are initialized according to $\lambda$ patterns observed in the data. To this end, we initialize the $n$th row of $W_o$ with pattern $\lambda_n$ (equation (4)), which is a vector corresponding to the $n$th label-assignment observed in the training data:

$$W_o^n = i(\lambda_n) \tag{4}$$

The initialization weight $i$ for unit $l$ of pattern $\lambda_n$ is set to 0 if the corresponding label is not present in the given instance; or to the upper bound $UB$ of the normalized initialization weights of hidden layer $h$ and output layer $o$, scaled by the number of times $c$ the pattern occurs in the data:

$$i(\lambda_n^l) = \begin{cases} \sqrt{c} \times UB & \text{if } \lambda_n^l = 1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$$UB = \frac{\sqrt{6}}{\sqrt{|h| + |o|}} \tag{6}$$

We follow Glorot and Bengio (2010) in using $\sqrt{6}$ as normalization factor for UB, and limit the number of patterns $\lambda$ to the most frequently observed label assignments.

Our model uses three types of features representing the screenplay's lexical make up, its underlying character relations, and interactions.

**Lexical Features** An obvious feature class is the language of the movie. Comedies will be characterized by a different vocabulary compared to thrillers or historical drama. We thus represent each script as a vector of 7,500 dimensions corresponding to the most frequent words in the training corpus. Vector components were set to the

words' tf-idf values. Words in scripts were further annotated with their sentiment values using the AFINN lexicon (Nielsen, 2011), a list of words scored with sentiment strength within the range $[-5,+5]$. We extracted several features based on these sentiment values such as the sentiment score of the entire movie, the number of scenes with positive/negative sentiment, the ratio of positive to negative scenes, and the minimum and maximum scene sentiment. From scene headings, we were also able to extrapolate the number of internal and external locations per script.

**Graph-based Features** Our graph-based features are similar to those described in Gorinski and Lapata (2015). Specifically, we view screenplays as weighted, undirected graphs, where vertices correspond to movie characters and edges denote character-to-character interactions (essentially the number of times two characters talk to each other or are involved in a common action). From the graph we extract features corresponding to the number of main and supporting characters, which we identify by measuring their centrality in the movie network (e.g., the number of edges terminating in a given node). We also estimate character polarity by summing the sentiment of each character's utterances as well as the ratio of positive to negative characters in a given script.

**Interaction-based Features** We extract features based on how often any two characters interact, i.e., whether they are engaged in a conversation or in the same event (e.g., if a character kills another). We identify interactions as described in Gorinski and Lapata (2015) and measure the number of interactions per scene and movie, the number of positive and negative interactions, and their ratio.

### 4.2 Movie Overview Decoder

Our decoder generates a movie overview from the multi-label encoding described above. For this, we adapt the LSTM architecture of Wen et al. (2015; 2016) which was originally designed for dialogue act generation (e.g., given input inform(type="hotel", count="182", dogsallowed="dontcare"), the network outputs "*there are 182 hotels if you do not care whether dogs are allowed*"). The network performs content selection, i.e., decides which attribute labels to talk about, while generating the sentences describing them.

As outlined in the lower part of Figure 2, a sigmoid control gate feeds a content vector, $p_0$, into a traditional LSTM cell to generate a natural language surface form. At each timestep $t$, the output word $w_t$ is drawn from an output distribution conditioned on the previous hidden layer $h_{t-1}$ *as well as* the previous content vector $p_{t-1}$. The content selection cell effectively acts as a *sentence planner*, retaining or omitting information from the original vector $p_0$ at every time step $t$ to guide the sentence generating LSTM cell. Our LSTM architecture is defined by the following equations:

$$i_t = \sigma(W_{wi}w_t + W_{hi}h_{t-1}) \quad (7)$$
$$f_t = \sigma(W_{wf}w_t + W_{hf}h_{t-1}) \quad (8)$$
$$o_t = \sigma(W_{wo}w_t + W_{ho}h_{t-1}) \quad (9)$$
$$\hat{c}_t = \tanh(W_{wc}w_t + W_{hc}h_{t-1}) \quad (10)$$
$$r_t = \sigma(W_{wr}w_t + W_{hr}h_{t-1}) \quad (11)$$
$$p_t = r_t \odot p_{t-1} \quad (12)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \hat{c}_t + \tanh(W_{pc}p_t) \quad (13)$$

where $\sigma$ is the sigmoid function, $i_t, f_t, o_t, r_t \in [0,1]^n$ are input, forget, output, and reading gates respectively, and $\hat{c}_t$ and $c_t$ are proposed cell value and true cell value at time $t$.

In the original paper, the input $p_0$ to the LSTM is a 1-hot representation of the information that should be included in the natural language output. In our setup, we relax this constraint such that each element of $p_0 \in [0,1]$, i.e., we directly use the output of the multi-label encoder just described.

### 4.3 Training

The proposed architecture is trained jointly in an end-to-end fashion, minimizing the objective:

$$F(\theta) = \sum_t y_t^T \log(\hat{y}_t) + ||p_T|| + \sum_{t=0}^{T-1} \eta\xi^{||p_{t+1}-p_t||} \quad (14)$$

where $y_t$ and $\hat{y}_t$ are the observed and predicted word distributions over the training data, $p_T$ is the content vector at the final time index $T$, $p_0$ is the initial content vector as given by the encoder network, and $\eta, \xi$ are training constants. The second term in the objective penalizes the network for generating output without realizing all required labels, while the third term deters the network from utilizing more than one label at any given time step.

The model is trained on pairs of scripts and sentences extracted from Jinni. To give a concrete example, a training instance for the *Plot* sentence from Figure 1 would consist of the features representing the movie's screenplay, and the overview's

| Attributes | ZeroR | NB | DS | SVM | Lib | MLE |
|---|---|---|---|---|---|---|
| Mood | 43.6 | 51.2 | 47.1 | 45.3 | 50.7 | **58.4** |
| Plot | 31.3 | 36.7 | 35.4 | 31.5 | 39.6 | **43.9** |
| Genre | 37.1 | 52.4 | 45.8 | 40.6 | 54.9 | **55.3** |
| Attitude | 63.0 | 68.5 | 67.3 | 64.0 | 71.6 | **76.5** |
| Place | 51.3 | 49.7 | 54.9 | 51.4 | 54.2 | **58.6** |
| Flag | 51.7 | 49.3 | 54.7 | 51.4 | 50.9 | **57.0** |
| All | 46.3 | 51.3 | 50.9 | 47.4 | 53.7 | **58.3** |

Table 2: Attribute identification (average %F1 across 10 folds). Best system in bold.

| Attributes | ZeroR | Lib | MLE |
|---|---|---|---|
| Mood | 43.4 | 48.0 | **61.6** |
| Plot | 32.8 | 38.9 | **42.9** |
| Genre | 37.8 | 54.6 | **58.5** |
| Attitude | 61.2 | 69.0 | **73.1** |
| Place | 48.2 | 46.1 | **54.4** |
| Flag | 48.8 | 46.4 | **54.0** |
| All | 45.4 | 50.5 | **57.4** |

Table 3: Attribute identification (%F1; test set).

*Plot* sentence "*The plot revolves around special agents, mind games, and a psychopath.*". The *Plot* multi-label network encodes the script into content vector $p_0$, and the LSTM learns which "labels" represented in $p_0$ to talk about while its training objective discourages to leave too many labels unmentioned. The observed output error is back-propagated through the LSTM and the embedding network using stochastic gradient descent (Bottou, 1991) with decaying learning rate.

## 5 Evaluation

In this section we report our evaluation experiments. We begin by assessing how good our encoder is at capturing screenplay content and then proceed to evaluated the generated overviews themselves.

### 5.1 How Good is the Encoder?

In order to assess encoder's ability to induce structure over screenplays, we focus solely on the top part of the architecture in Figure 2. Specifically, we trained stand-alone models for the six attributes shown in Table 1 on the gold data provided in the Jinni dataset. All networks used the same features introduced earlier and were initialized using the pattern-based method of Kurata et al. (2016). To better capture the fact that we are dealing with multi-label assignments, we used the global error function described in Zhang and Zhou (2006). Given the network output vector $\hat{y}$ for input $x$, the true bag of label assignments $y$ and its complement $\bar{y}$, the error observed for each instance is computed as:

$$E = \frac{1}{|y||\bar{y}|} \sum_{(k,l)\in y\times\bar{y}} \exp(-(\hat{y}_k - \hat{y}_l)) \qquad (15)$$

The networks were trained with stochastic gradient descent during back propagation, using the same method as for the full model.

We compared our multi-label encoders (MLE) against several baselines. These include assigning the most frequent attribute labels to each movie based on the attributes' mean distribution (ZeroR), Naive Bayes (NB), Decision Stump (DS), LibLinear (Lib; Fan et al., 2008) and Support Vector Machines (SVMs; Chang and Lin, 2011). For each comparison system, we trained a binary classifier per attribute label using features identical to the ones used for the MLE.

Table 2 shows F1 performance on the training data for MLE and comparison systems, averaged over 10 folds. As can be seen, MLE performs best, followed by LibLinear. Table 3 compares MLE, ZeroR, and Lib, the strongest baseline, on the test set using F1 and the best parameters found for each system during cross-validation. As can be seen, MLE outperforms Lib across attributes, and is superior to ZeroR by a large margin. F1 differences between MLE and LibLinear are significant ($p < 0.01$), using approximate randomization testing (Noreen, 1989).

Overall, the results in Tables 2 and 3 indicate that the classification task is hard. This is especially true for *Plot* which has the largest number of labels. Nevertheless, the multi-label encoders introduced here achieve good performance on their own, indicating that they are able to capture the content of the screenplay, albeit approximately.

### 5.2 How Good is the Decoder?

We next evaluate the performance of the jointly trained system which we call MORGAN as a shorthand for **M**ovie **O**ve**R**view **G**ener**A**tio**N** model. MORGAN is trained on pairs of screenplays and their corresponding verbalizations in the Jinni dataset. Unfortunately, our dataset is relatively small for neural network training; it contains 617 movies only, i.e., there are 617 sentences for each attribute. To alleviate this problem, we augmented the data as follows. We extracted sentence templates from the training set (209 in total), ex-

| | |
|---|---|
| Mood | $T$ can be described as $M_1$ and $M_2$<br>The mood of $T$ is $M_1$. |
| Plot | The plot centers around a $P_1$, $P_2$, and $P_3$<br>The plot revolves around $P_1$, $P_2$, and $P_3$ |
| Genre | The main genres are $G_1$, $G_2$, and $G_2$<br>$T$ is $M_1$ and $M_2$ movie |
| Attitude | In approach, $T$ is $A_1$<br>The pacing is $A_1$ |
| Place | The setting is $L_1$<br>It is located in $L_1$ |
| Flag | Note that the movie involves $F_1$ and $F_2$<br>Note that it includes $F_1$, $F_2$, and $F_3$ |

Table 4: Template sentences extracted from Jinni overviews. Variable $T$ is filled by the movie's title, whereas $M$, $G$, $P$, $A$, $L$, $F$ correspond to values for attributes *Mood*, *Genre*, *Plot*, *Attitude*, *Place*, and *Flag*, respectively.

amples of which are shown in Table 5. We replaced the title and attribute values with variables (shown as capital letters in the table). We then used the templates to generate additional data for each movie by substituting attribute variables in template sentences with permutations of the movie's gold-standard labels. We thereby obtained a total of 31,000 training instances.

The model was trained with a learning rate of 0.5, using a decay of 0.01 over 50 epochs, fixing it for subsequent epochs. Constants $\eta$ and $\xi$ in equation (14) were set to $10^{-4}$ and 100, respectively. At test time, we used screenplay features as input and generated one sentence per attribute. We arranged these into an overview following the ordering *Mood* $\gg$ *Plot* $\gg$ *Genre* $\gg$ *Attitude* $\gg$ *Place* $\gg$ *Flag* which is fixed and attested in all overviews in our dataset.

We compared MORGAN against several systems: (1) a random baseline, selecting for each movie and attribute type a random sentence from the training set; (2) a nearest-neighbor baseline (NN) which uses the same screenplay features as MORGAN (and cosine similarity) to identify the closest matching script in the training data, and rehashes its overview as output; (3) an attention-based LSTM (Bahdanau et al., 2015) trained on script sentence pairs (31,000 in total); and (4) six attention-based LSTMs, one per attribute type, trained on script sentence pairs (on average 5,200 per LSTM). The attention LSTMs were trained on the same screenplay features as MORGAN, with the attention mechanism at each timestep $t$ focusing on parts of the input. Example overviews generated by MORGAN, the attention LSTMs, and the nearest neighbor system are shown in Table 6.

| Models | BLUE | Coherence | Grammaticality |
|---|---|---|---|
| Random | 38.0 | 2.42* | 3.83 |
| NN | 40.4 | 3.45 | 3.93 |
| Attn | 23.0 | 2.93* | 3.91 |
| typAttn | 37.9 | 3.20 | 3.80 |
| MORGAN | **42.0** | **3.72** | **4.08** |
| Jinni | — | 4.27 | 4.22 |

Table 5: BLEU scores and mean coherence and grammaticality ratings for movie overviews. * significantly different from MORGAN ($p < 0.05$). Best performing system shown in bold.

We evaluated system output with multi-reference BLEU[4] (Papineni et al., 2002), using sentences from the extended gold-standard as references. Table 5 (first column) summarizes our results. As can be seen, MORGAN outperforms the attention based models, the nearest neighbor system, and the random baseline. The attention-based models cannot succinctly capture the movie's content in order to render it into meaningful sentences. Although the generated sentences are more or less grammatical on their own (see Table 6), the generated overview lacks coherence, and is fairly repetitive. The model does not reliably learn what type of information to focus on for the generation task. For MORGAN this problem is alleviated during the encoding step, which performs content distillation prior to generating overview sentences.

### 5.3 How are System Overviews Perceived by Humans?

In addition to evaluating system output automatically, we are also interested in how it is perceived by humans. To this end, we ran two judgment elicitation studies on Amazon Mechanical Turk. Both experiments were conducted on 12 movies. In a pre-test we asked 20 workers whether they had seen the movies in our test set and chose the three most popular ones from each of the genres Action, Comedy, Drama, and Romance.

In our first experiment Turkers were presented with an overview taken from the Jinni gold standard, MORGAN or one of the comparison systems and asked to rate its coherence (i.e., whether it was readily comprehensible or difficult to follow) on a scale from 1 (incoherent) to 5 (coherent). Subsequently, they had to rate the grammaticality of

[4]We use NLTK's (http://www.nltk.org/) implementation of BLEU, and report the interpolation of BLEU 1 through 4.

| Nearest Neighbor |
|---|
| The mood of Burn after Reading is stylized. The plot revolves around love and romance, motives, buddies, and friends. Its main genres are romance, crime, and comedy. In approach, Burn after Reading is semi serious. It takes place, at least partly, in the USA. Note that Burn after Reading involves violent content, profanity, and sexual content. |
| attnLSTM |
| The setting is Europe. It is a sci-fi, adventure, and adventure movie. Note that Burn after Reading includes violent content. Burn after Reading is set in the USA. It is located in the USA. It is set, at least in part, in the USA. |
| typed attnLSTM |
| The mood of Burn after Reading is bleak, touching, and feel good. The plot revolves around ambition motives, love and romance, and fall in love. It is an adventure and fantasy, and features thriller themes. In approach, Burn after Reading is serious. Burn after Reading is set, at least in part, in the USA. Note that Burn after Reading involves violent content. |
| MORGAN |
| The mood of Burn after Reading is feel good, exciting, and clever. The plot centers around ambition, special agents, and partners. Burn after Reading is a comedy movie. In approach, it is realistic and semi serious. It is set, at least in part, in the USA. Note that it involves profanity. |
| JINNI |
| Burn after Reading can be described as cynical, clever, and exciting. The plot revolves around espionage , disorder, and escapades. Burn after Reading's main genres are comedy and crime. In approach, it is not serious and realistic. Burn after reading is set, at least in part, in an urban environment. It is located in Virginia. Note that it involves sexual content, profanity, and violent content. |

| Nearest Neighbor |
|---|
| Lara Croft: Tomb Raider can be described as touching, offbeat, and humorous. The plot revolves around a psychopath, a chase, and couples motives. Its main genres are drama and comedy. In approach, Lara Croft: Tomb Raider is semi serious. Lara Croft: Tomb Raider is set in the USA. |
| attnLSTM |
| Lara Croft: Tomb Raider is set, at least in part, in a dangerous and in a haunted house. The plot centers around buddies, disorder, and criminal heroes motives. The plot centers around escapades, supernatural, and deadly. The setting is a school or college environment. Note that Lara Croft: Tomb raider includes drugs/alcohol and sexual content. The mood of Lara Croft: Tomb Raider is rough, being, and contemplative. |
| typed attnLSTM |
| The mood of Lara Croft: Tomb Raider is suspenseful, exciting, and stylized. The plot revolves around a police investigation, cons and scams, and couple relations motives. Lara Croft: Tomb Raider's main genres are fantasy, action, and comedy. In approach, it is serious and fantastic. Lara Croft: Tomb Raider is set, at least in part, in the USA. Note that Lara Croft: Tomb Raider involves profanity and violent content. |
| MORGAN |
| Lara Croft: Tomb Raider can be described as suspenseful and rough. The plot centers around danger, a master villain, and deadly. Lara Croft: Tomb raider is an adventure, thriller, and action movie. In approach, it is fantastic. Lara Croft: Tomb Raider takes place, at least partly in Europe. Note that it involves profanity. |
| JINNI |
| Lara Croft: Tomb Raider can be described as stylized, exciting, and suspenseful. The plot revolves around time travel, a treasure hunt, and a master villain. The main genres are action, adventure, and fantasy. In approach, it is fantastical and serious. Note that it involves violent content. |

Table 6: Overviews generated by MORGAN and comparison systems for "Burn after Reading" (top) and "Lara Croft; Tomb Raider" (bottom).

each overview sentence, again on a scale from 1 (ungrammatical) to 5 (grammatical) and decide whether it appropriately described aspects of the movie's content ("Yes", "No", "Unsure"). We elicited five responses for each overview across six systems (Jinni, typAttn, Attn, Random, NN, and MORGAN) and 12 movies. Finally, participants had to answer a question relating to the movie's content, to make sure that they had actually seen the movie. We discarded responses with wrong answers to the content question. Examples of the overviews participants judged are given in Table 6.

Table 5 (columns 2 and 3) summarizes the results of our first judgment elicitation study. All systems perform well with regards to grammaticality. This is not surprising for Random and NN which do not perform any generation. Attn and typAttn also perform well with MORGAN achiev-

| Model | Mood | Plot | Genre | Attitude | Place | Flag | All |
|---|---|---|---|---|---|---|---|
| Random | 37.7 | 39.6 | 34.0 | 43.4 | 35.8 | 50.9 | 19.0* |
| NN | 78.6 | 67.9 | 71.4 | 66.1 | 58.9 | 91.1 | 58.9* |
| Attn | 38.2 | 38.2 | 38.2 | 41.8 | 51.0 | 34.5 | 40.0* |
| typAttn | 60.0 | 60.0 | 53.3 | 57.8 | **66.7** | 64.4 | 40.0* |
| MORGAN | **89.5** | **73.7** | **80.7** | **71.9** | 63.2 | **89.5** | 82.5 |
| Jinni | 91.1 | 89.3 | 92.9 | 82.1 | 67.9 | 75.0 | 91.1 |

Table 7: Proportion of sentences and overviews (All) which describe the movie accurately. * significantly different from MORGAN ($p < 0.05$). Best performing system per attribute is in bold.

ing highest scores for grammaticality amongst automatic systems. Grammaticality differences between the various systems in Table 5 and the Jinni gold standard are not statistically significant (using a one-way ANOVA with post-hoc Tukey HSD tests). Overviews generated by MORGAN are perceived as more coherent in relation to those generated by comparison systems, even though the model does not explicitly take coherence into account. MORGAN overviews are not significantly different in terms of coherence from Jinni, typAttn, and NN, but are significantly better than Random and Attn.

Table 7 shows the percentage of sentences (per attribute and overall) which participants think describe the movie's content felicitously. MORGAN identifies most aspects of the movie successfully, in some cases close to (*Mood, Place*) or even better (*Flag*) than the original Jinni overview. MORGAN is significantly better compared to all other models but not significantly worse than Jinni (using a $\chi^2$ test; see last column in Table 7).

In a second experiment, participants were presented with six overviews for a movie (from Jinni, Attn, typAttn, Random, NN, and MORGAN) and asked to rank them (equal ranks were not allowed) in order of relevance (i.e., whether they express content relevant to the movie). Again, we obtained five responses for each movie. As can be seen in Table 8, while Jinni is ranked first most of the time, MORGAN is ranked second followed by the NN system. We further converted the ranks to ratings on a scale of 1 to 6 (assigning ratings 6...1 to rank placements 1...6) and performed and ANOVA which showed that all systems are significantly ($p < 0.05$) worse than Jinni but MORGAN is significantly better than the comparison systems.

| Model | 1st | 2nd | 3rd | 4th | 5th | 6th | AvgRank |
|---|---|---|---|---|---|---|---|
| Random | 1.0 | 5.8 | 16.3 | 22.1 | 19.2 | **35.6** | 4.59 |
| NN | 5.8 | 19.2 | **24.0** | 23.1 | 15.4 | 12.5 | 3.60 |
| Attn | 3.8 | 13.5 | 20.2 | **28.8** | 16.3 | 17.3 | 3.92 |
| typAttn | 1.9 | 7.7 | 15.4 | 10.6 | **33.6** | 30.8 | 4.58 |
| MORGAN | 8.7 | **42.3** | 22.1 | 12.5 | 12.5 | 1.9 | 2.71 |
| Jinni | **78.8** | 11.5 | 1.9 | 2.9 | 2.9 | 1.9 | 1.45 |

Table 8: Relevance rankings (shown as proportions) given to overviews by human subjects. Most frequent rank per system and Jinni is in bold.

## 6 Conclusions

In this work we have presented a novel approach to automatic movie content analysis. We have assembled a new dataset which combines ScriptBase (Gorinski and Lapata, 2015), a corpus of movie scripts, with information gathered from Jinni, a large movie database. We proposed an end-to-end model for movie overview generation via *multi-attribute* encoders and a *semantically conditioned* LSTM decoder. Experimental results show that our encoders are capable of distilling meaningful structures from the screenplay. When applied to the overview generation task, our end-to-end model outperforms a standard attention-based LSTM. Human evaluation also indicates the overviews generated by our model are felicitous, informative, and rated favorably by humans.

In the future, we would like to investigate how attribute-specific features can improve performance compared to our more general feature set which is invariant for each sentence type. It would also be possible to equip the model with a hierarchical decoder which generates a document instead of individual sentences. Although currently our model relies solely on textual information, it would be interesting to incorporate additional modalities such as video (Zhou et al., 2010) or audio (e.g., we expect comedies to be *visually* very different from thrillers, or romantic movies to have a different *score* from superhero movies). Finally, we would like to examine whether the content analysis presented here can extend to different types of fiction such as novels or short stories.

**Acknowledgments** We thank the NAACL reviewers for their constructive feedback. We gratefully acknowledge the financial support of the European Research Council (award number 681760).

## References

Apoorv Agarwal, Sriramkumar Balasubramanian, Jiehan Zheng, and Sarthak Dash. 2014a. Parsing Screenplays for Extracting Social Networks from Movies. In *Proceedings of the 3rd Workshop on Computational Linguistics for Literature*. Gothenburg, Sweden, pages 50–58.

Apoorv Agarwal, Sarthak Dash, Sriramkumar Balasubramanian, and Jiehan Zheng. 2014b. Using Determinantal Point Processes for Clustering with Application to Automatically Generating and Drawing xkcd Movie Narrative Charts. In *Proceedings of the 2nd Academy of Science and Engineering International Conference on Big Data Science and Computing*. Stan-

ford, California.

Apoorv Agarwal, Jiehan Zheng, Shruti Kamath, Sriramkumar Balasubramanian, and Shirin Ann Dey. 2015. Key Female Characters in Film Have More to Talk About Besides Men: Automating the Bechdel Test. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado, pages 830–840.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*. San Diego, CA.

David Bamman, Brendan O'Connor, and Noah A. Smith. 2013. Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Sofia, Bulgaria, pages 352–361.

David Belanger and Andrew McCallum. 2016. Structured prediction energy networks. In *Proceedings of the 33rd International Conference on Machine Learning*. New York, pages 983–992.

Léon Bottou. 1991. Stochastic gradient learning in neural networks. In *Proceedings of the 7th Neuro-Nîmes International Conference*. EC2, Nîmes, France.

Chih-Chung Chang and Chih-Jen Lin. 2011. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2(3):27.

Liang-Chieh Chen, Alexander G Schwing, Alan L Yuille, and Raquel Urtasun. 2015. Learning deep structured models. In *Proceedings of the 32nd International Comference on Machine Learning*. Lille, France, pages 1785–1794.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar, pages 103–111.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of NAACL: HLT*. San Diego, CA, pages 93–98.

Mita K. Dalal and Mukesh A. Zaveri. 2011. Automatic Text Classification: A Technical Review. *International Journal of Computer Applications* 28(2):37–40.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in Imagined Conversations: A New Approach to Understanding Coordination of Linguistic Style in Dialogs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*. Portland, Oregon, pages 76–87.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research* 9 (Aug):1871–1974.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*. Chia Laguna Resort, Sardinia, Italy, pages 249–256.

Shantanu Godbole and Sunita Sarawagi. 2004. Discriminative methods for multi-labeled classification. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, pages 22–30.

Philip John Gorinski and Mirella Lapata. 2015. Movie script summarization as graph-based scene extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado, pages 1066–1076.

John R. Hershey, Jonathan Le Roux, and Felix Weninger. 2014. Deep unfolding: Model-based inspiration of novel deep architectures. *CoRR* abs/1409.2574.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR* abs/1508.01991.

Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep structured output learning for unconstrained text recognition. *CoRR* abs/1412.5903.

Gakuto Kurata, Bing Xiang, and Bowen Zhou. 2016. Improved neural network-based multi-label classification with better initialization leveraging label co-occurrence. In *Proceedings of the 2016 Conference of the North Amer-*

*ican Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California, pages 521–526.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pages 1203–1213.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California, pages 720–730.

Finn A. Nielsen. 2011. A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the ESWC2011 Workshop on Making Sense of Microposts: Big things come in small packages*. Heraklion, Crete, pages 93–98.

Eric Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses: An Introduction*. Wiley, Hoboken, New Jesey.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA, pages 311–318.

Jesse Read, Bernhard Pfahringer, and Geoff Holmes. 2008. Multi-label classification using ensembles of pruned sets. In *Proceedings of the 8th International Conference on Data Mining*. Pisa, Italy, pages 995–1000.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2011. Classifier chains for multi-label classification. *Machine Learning* 85(3):333–359.

Alexander G. Schwing and Raquel Urtasun. 2015. Fully connected deep structured networks. *CoRR* abs/1503.02351.

Fabrizio Sebastiani. 2002. Machine Learning in Automated Text Categorization. *Association for Computing Machinery: Computing Surveys* 34(1):1–47.

Veselin Stoyanov, Alexander Ropson, and Jason Eisner. 2011. Empirical risk minimization of graphical model parameters given approximate inference, decoding, and model structure. In *AISTATS*. pages 725–733.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*. pages 3104–3112.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 1422–1432.

Grigorios Tsoumakas and Ioannis Katakis. 2006. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining* 3(3).

Grigorios Tsoumakas and Ioannis Vlahavas. 2007. Random k-labelsets: An ensemble method for multilabel classification. In *European Conference on Machine Learning*. Springer, pages 406–417.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 3156–3164.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pages 1711–1721.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, and Steve Young. 2016. Multi-domain neural network language generation for spoken dialogue systems. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California, pages 120–129.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 1480–1489.

Patrick Ye and Timothy Baldwin. 2008. Towards Automatic Animated Storyboarding. In

*Proceedings of the 23rd Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence*. Chicago, Illinois, pages 578–583.

Min-Ling Zhang and Zhi-Hua Zhou. 2005. A k-nearest neighbor based algorithm for multi-label classification. In *Proceedings of the IEEE International Conference on Granular Computing*. Beijing, China, volume 2, pages 718–721.

Min-Ling Zhang and Zhi-Hua Zhou. 2006. Multi-label neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering* 18(10):1338–1351.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark, pages 595–605.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. 2015. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*. Santiago, Chile, pages 1529–1537.

Howard Zhou, Tucker Hermans, Asmita V. Karandikar, and James M. Rehg. 2010. Movie genre classification via scene categorization. In *Proceedings of the 18th ACM International Conference on Multimedia*. New York, NY, pages 747–750.